BLACK CODHER

CODING PROGRAMME

CODING BLACK FEMALES

Black Codher Bootcamp

NÍ YÓ

BLACK CODHER

CODING PROGRAMME

UNIT 4 - Session 4
React

End of Session 3 Summary

In the last session we covered the following:

1. Editing the Create React App
2. Viewing the React elements in a browser window (using Developer Tools)
3. Class components and Functional Components
4. JSX Components Explained
5. Understanding the Component Lifecycle
6. Understanding Props (Properties) and States in React

Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
3

Goals for Unit 4 - Session 4

1. Understanding React Hooks
   a. useState
   b. useEffect

2. Understanding JSON files

3. Understand Object and Array deconstruction

4. Creating your first JSX component

5. Start building a Library/Bookcase React App

Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
4

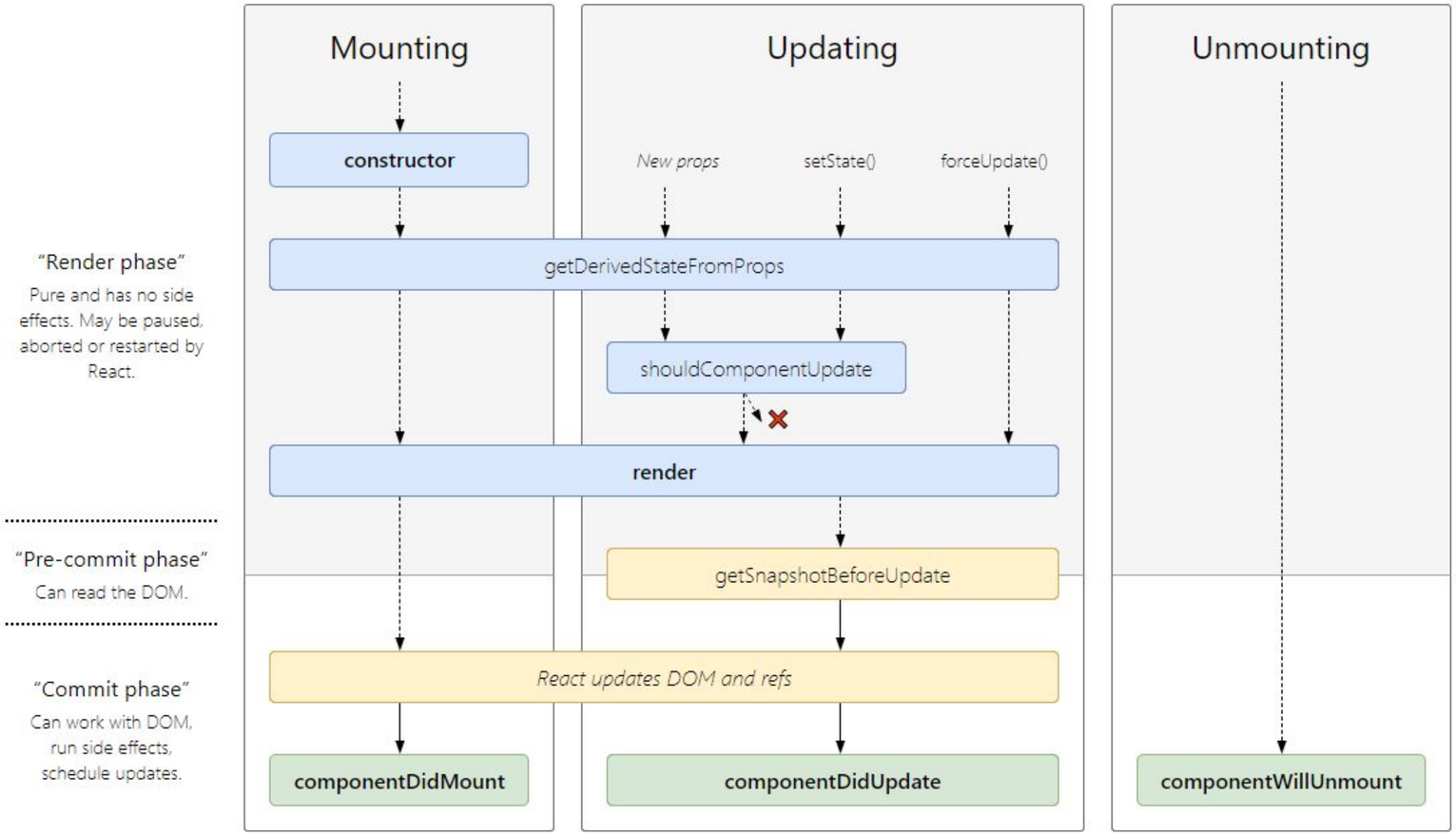
React Hooks

5
Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

Lifecycle Cheatsheet Reminder

- React Hooks allow you to “hook onto” state and lifecycle features without writing a class component

- With class components state and lifecycle methods are inherited from a parent class and **overridden*** by a child class



Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
6

React Hooks

BLACK CODHER
CODING PROGRAMME

- **Hooks** are a new addition in React 16.8 (released in Feb 2019).
- They let you use state and other React features without writing a class.
- Hooks are special functions that allow you to 'hook onto' React state and lifecycle features inside a **functional component**.
- Hooks are backward compatible, so, It is possible to convert your **class components** into **functional components** that use Hooks.
- React Hooks are completely **opt-in**, you can try Hooks in a few components without rewriting existing code.

7
Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

React Hooks

- They are also 100% backwards compatible.
- There are no plans to remove class components from React.
- Hooks do not replace knowledge of React concepts.

8
Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

React Hooks

BLACK CODHER
CODING PROGRAMME

The main rules for using hooks are:

1. They should only be called at the top level (not inside loops, conditions or nested functions)

2. Hooks should only be called from React functional components (don't call hooks from regular JavaScript functions)

- These rules are to ensure that stateful logic is clearly visible.

9
Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

State Hook Example: useState()

- To set state using a React Hook, you would use the following syntax in your functional component:

```
const [count, setCount] = useState(0);
```

- This example above uses **array destructuring*** to set the variable **count** with a default of **0** and a method of **setCount** for updating the variable

- It is equivalent to this code:

```
const countStateVariable = useState(0);
const count = countStateVariable[0];
const setCount = countStateVariable[1];
```


Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
10

State Hook Example: useState()

Here is an example of destructuring in a functional BookCounter component

```
const BookCounter = (props) => {
const [count, setCount] = useState(0);

return (
   <div className="booklist">
   <h1>{props.library.name}'s Books ({count})  &#8595;</h1>
   <button onClick={() => setCount(count + 1)}>Count Books</button>
     <ul>
       ...
     </ul>
    </div>
  );}
```


Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
11

State Hook Example: useEffect()

- The useEffect() Hook tells your component to do something after every **render**.

```
useEffect(() => {
    document.title = `${count} Book(s) counted`
});
```

- The code in **useEffect** above would be called every time a component is rendered.

- This would means that the title tag of the site will change on each render if the variable count has changed

Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
12

State Hook Example: useEffect()

BLACK CODHER
CODING PROGRAMME

- **useEffect** is placed inside a functional component and is equivalent to **componentDidMount**(), **componentDidUpdate**() and **componentWillUnMount**() all in one

- **useEffect**() takes two arguments. The first is the function to call and the second argument is an array which can be used to define how many times the first argument should be called

- There are other hooks that are less commonly used such as **useContex**() and **useReducer**()

- For more information on all the Hooks in React: https://reactjs.org/docs/hooks-reference.html#gatsby-focus-wrapper

13
Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

Checkpoint!

How are you feeling?

RED - I have no idea what you're talking about.

YELLOW - I have some questions but feel like I understand some things.

GREEN - I feel comfortable with everything you've said.

Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

Exercise 1

Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
15

Exercise 1: useState Hook

BLACK CODHER
CODING PROGRAMME

1. Navigate to the **`black-codher-bootcamp`** directory on your machine

```
> cd black-codher-bootcamp
```

2. Get the latest files from the repository

```
> git pull
```

3. Copy the file **BookCounter.js** from the folder **black-codher-bootcamp\unit04-react\session4** into your **test-project\src** directory.

4. In Visual Code open the **index.js** file in your **test-project** and overwrite the content with the following lines of code:

Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
16

Exercise 1: useState Hook

```
import React, {Fragment} from 'react';
import ReactDOM from 'react-dom';

import BookCounter from './BookCounter';

const element = <Fragment>
    <h1>Welcome to My Library</h1>
    <BookCounter library={{name:"Sarah",theme:"Modern"}}/>
</Fragment>;

ReactDOM.render(element,document.getElementById('root'));
```

5. Run your project (**> npm start**)

6. Look through the lines of code in BookCounter.js

Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

Exercise 1: useState Hook

Output of exercise:

- Clicking the “Count Books” button will increment the book counter

Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
18

Exercise 1: useState Hook

BLACK CODHER
CODING PROGRAMME

BookCounter.js:

- The BookCounter.js file contains a functional component called **BookCounter**. It's declared with the following syntax:

```
const BookCounter = (props) => { ... }
```

- When the button rendered to the page is clicked, an internal counter is incremented using the **useState** hook (line 5 and 10 of BookCounter.js)

Index.js:

- Line 22 exports the BookCounter making it available to the rest of the project through the import syntax (line 4 of index.js)

- Properties (props) are declared in the attribute of the BookCounter (line 8 of index.js)

19
Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

Exercise 1: useState Hook

BLACK CODHER
CODING PROGRAMME

- It's also interesting to note how the BookCounter is declared in **index.js** using JSX (line 8):

```
<BookCounter library={{name:"Sarah",theme:"Modern"}}/>
```

- It's important to note that the library attribute is a Javascript object

- The attribute (library) is then available/used in the BookCounter component (line 9 of BookCounter.js)

7. Open the file BookCounter.js. It contains a functional component called BookCounter. Add some style to the BookCounter component's **h1**, **button** and list of books (**li**). Start by creating a file called **BookCounter.css**.

20
Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

Exercise 1: useState Hook

8. Add some style to the BookCounter component's **h1**, **button** and list of books (**li**). Start by creating a file called **BookCounter.css**.

9. Add a class called **booklist** to the **BookCounter.css** file (e.g, .booklist { ... })

10. Add the following line at the top of the BookCounter.js file:

```
import './BookCounter.css';
```


Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
21

A little about JSON...

22
Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

JSON

- JSON stands for JavaScript Object Notation
- JSON objects can be used for transferring data, XML* serves the same purpose. However XML is verbose and can get very large.
- JSON objects have several advantages over XML:
  - They are lightweight
  - Easy to write
  - Text based and human readable
- JSON is considered a subset of JavaScript

Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

My Bookcase Application

- JSON follows these syntax rules:
  - Data is in name/value pairs
  - Data is separated by commas
  - Curly braces hold objects
  - Square brackets hold arrays

An example of an array of JSON people objects, containing attributes of name, age and city:

```
[
  { name: "Jessica", age: 52, city: "New York"},
  { name: "Simone", age: 45, city: "London"},
  { name: "Zoe", age: 19, city: "Amsterdam"},
  { name: "Faith", age: 28, city: "Berlin"},
]
```


Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
24

Destructuring

25
Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

Destructuring Assignment

BLACK CODHER
CODING PROGRAMME

- Objects allow us to create a single entity that holds data items by keys and arrays allow to hold data items in a collection.

- **Destructuring** assignment is a syntax in ES6 that allows us to unpack an array or objects into variables .

- It is great for reducing the complexity of code and shortening the dot notation needed to reference attributes in a complex objects.

26
Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

Destructuring Assignment

Array Destructuring:

```
let [firstname, lastname] = "Jackie Christensen".split(' ');
/* Equivalent to:
let firstname = names[0];
let lastname = names[1]; */
```

- It's also possible to skip unwanted parts of the array:

```
let [firstname, ,occupation] = ["Angelica","Smith","Data
Analyst","Brentford"];
alert(occupation); //Data Analyst
```


Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
27

Destructuring Assignment

- Or set default values:

```
let [firstname, lastname = "Jackson"] = "Jackie";
alert(firstname + ' ' + lastname); //Jackie Jackson
```

Object Destructuring: Example of a book object

```
book: {
   "id": "djc_DwAAQBAJ",
    "volumeInfo": {
      "title": "The Girl Who Smiled Beads",
      "authors": ["Clemantine Wamariya"]
       "description": "When Clemantine Wamariya was six years old..."
     },
    "price":9.99
}
```


Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

Destructuring Assignment

- Variables can be referenced in the following way:

```
let {id, price} = book;
```

Nested Object Deconstructuring:

```
let {id, volumeInfo: {title, authors, description}, price} = book;

alert(id);  //book.id
alert(title); //as oppose to book.volumeInfo.title
```


29
Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

Exercise 2

Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
30

Exercise 2: Destructuring

1. Open your React test project

2. Copy the file **books.json** file from **black-codher-bootcamp\unit04-react\session4** to the **src** folder

3. Copy the code on the following slide into the **index.js**

Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
31

Exercise 2: Destructuring

```
import React, {Fragment} from 'react';
import ReactDOM from 'react-dom';
import books from './books.json';

const formatter = new Intl.NumberFormat('en-GB', {
  style: 'currency',
  currency: 'GBP'
})

const book = books[0];
let {id, volumeInfo: {title, authors, description}, saleInfo: {listPrice: {amount}}} = book;

const element = <Fragment>
    <h1 id={id}>{title} = {formatter.format(amount)}</h1>
</Fragment>;

ReactDOM.render(element,document.getElementById('root'));
```


Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
32

Exercise 2: Destructuring

BLACK CODHER
CODING PROGRAMME

- The script displays the first book in the **books.json** file
- Note how the books.json is imported as a data source into the books variable (line 2)
- The variable book is set to the first object in the book array on line 10
- The object is deconstructed on line 11 into several variables
- The book is displayed as an JSX element on line 13

4. Write a for-loop to loop through all the books in the books list and display the results of the title, description and price of a books

33
Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

Exercise 2: Destructuring

5. Display the author(s), it's important to note that the authors field is an array e.g. {

```
...
"authors": ["Clemantine Wamariya"]
...
}
```

6. Look through the **books.json** file and identify an image attribute (*hint: you can use either the smallThumbnail or thumbnail attributes*). Add the image attribute to the deconstructed variables on line 11 of **index.js**

7. Display the book images using an `<img src='' alt=''/>` HTML element

34
Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

Create a Bookcase Application

Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
35

Building a React Application

By the end of the React module we are going to build a full application for searching and storing books that you have read and want to read. The application will be called My Library and it will:

1. Load an initial set of suggested book from a local data store (JSON file)

2. Add a book to a bookcase of books

3. Allow a user to navigate between a search screen and their bookcase of books. (Add routing)

4. The book generator will link to a third-party API provided by Google (Books APIv1: https://developers.google.com/books/docs/overview)

36
Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

Building a React Application

5. Allow a user to view an about us page explaining the about us
6. Keep a count of the number of books in the bookcase and display in the document.title and on the page
7. Add a search bar form
8. Search the API to display books relevant to the query
9. Allow a user to browse a library of books by book name, author or theme
10. Remove a book from a bookcase of books

37
Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

Building a React Application

11. Add pagination (next and previous button)

12. Advanced Pagination: Add numbered pages and result display

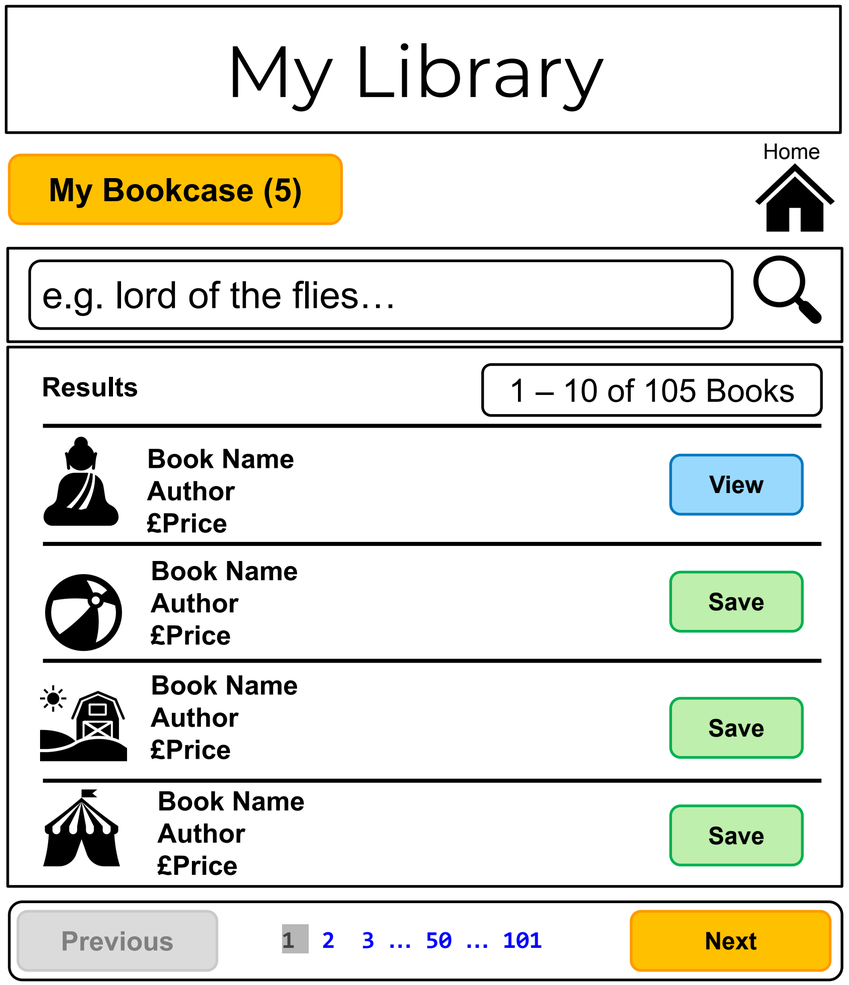
Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
38

Exercise 3

Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
39

Exercise 3: My Library Components

Separate and highlight the possible components of this interface or design your own:



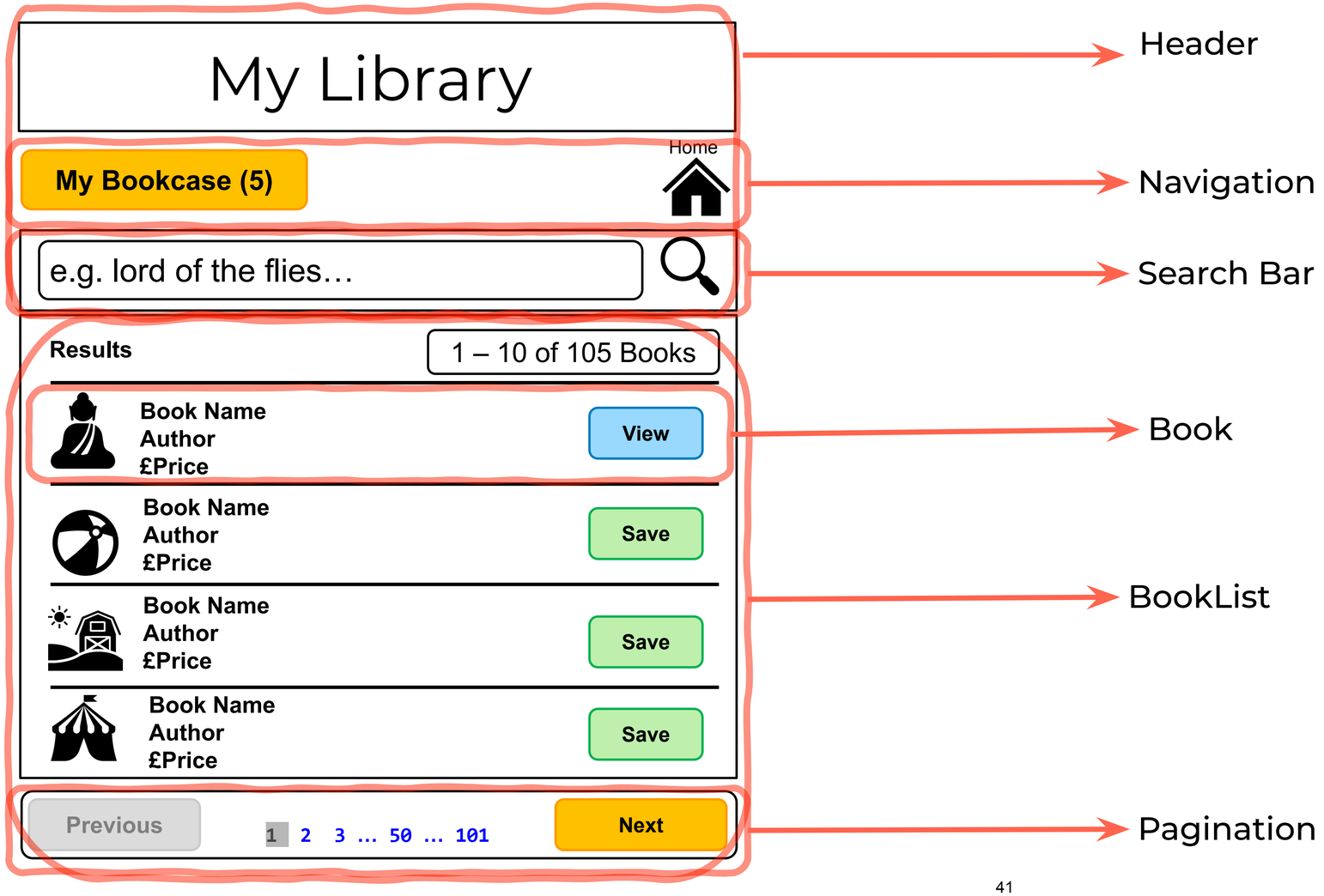
Some example components:

- Header
- Search Bar
- Pagination
- Book
- BookList
- Navigation

Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

40

User Interface to Components



Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

Exercise 3: My Library Application

BLACK CODHER
CODING PROGRAMME

- In Visual Code create a new folder called **mybookcase**, or type into the terminal

```
> mkdir mybookcase
> cd mybookcase
```

- In the Visual Code terminal type:

```
> npx create-react-app .
```

- The full stop in this case just means create the app in the current folder

```
Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts with cra-template...

[...................] / fetchMetadata: sill resolveWithNewModule scheduler@0.19.1 checking installable status
```


Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
42

Exercise 3: Application Folders

BLACK CODHER
CODING PROGRAMME

- Reset the app by removing all the files in the **src** folder except **index.js** and **App.js**.

- Create three new folders in the **src** directory called **components**, **models** and **pages**. Or type into the terminal:

```
> mkdir components,models,pages
```

- The app will use a JSON file as it's initial source of data. The JSON file **books.json** can be found in the **black-codher-bootcamp/unit04/session4** folder. Get the latest file by navigating to the **black-codher-bootcamp** folder and running the following git command from the terminal:

```
> git pull
```

- Copy the JSON file **books.json** into the **models** folder

43
Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

Books.json Application Data

- The initial structure of our app will be as follows:

```
<App>
    <Book/>
    <Book/>
    <Book/>
    <Book/>
    //... Multiple <Book/> elements
</App>
```

- Each <Book> tag is a JSX element and will be a functional component defined in a file called Book.js

- An example Book.js file is detailed in the homework assignment

Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
44

Session 4 Summary

1. Understanding React Hooks
   a. useState
   b. useEffect

2. Understanding JSON files

3. Understand Object and Array deconstruction

4. Creating your first JSX component

5. Start building a Library/Bookcase React App
   a. Identified the components in the App
   b. Built a Book component

Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
45

Checkpoint!

How are you feeling?

RED - I have no idea what you're talking about.

YELLOW - I have some questions but feel like I understand some things.

GREEN - I feel comfortable with everything you've said.

Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

Homework: My Library App

Convert the list of books from **book.json** into a list of JSX components.

1. Copy the **books.json** into the **models** folder of your **mybookcase** app (slide 40)

2. Update your **index.js** with the following code

```
import React from 'react';
import ReactDOM from 'react-dom';
import App from './App';

ReactDOM.render(
  <React.StrictMode>
    <App />,
  </React.StrictMode>,
  document.getElementById('root')
);
```


Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
47

Homework: My Library App

3. Add a new file called **Books.js** to the **components** directory with the following code:

```
import React from 'react';

const Book = (props) => {
    return (
          <div>
            <h2>{props.book.volumeInfo.title}</h2>
          </div>
    );
}
export default Book;
```


Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
48

Books.json Application Data

4. Add the following code to your App.js:

```
import React, { useState } from 'react';
import Book from './components/Book';
import data from './models/books.json';

const App = (props) => {

  const [books] = useState(data);

  return (
      <div>
          {books.map(book => <Book key={book.id} book={book}/>)}
      </div>
  );
}

export default App;
```


49
Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.

Books.json Application Data

5. Update the code in Book.js to add object deconstruction, follow the steps outlined in exercise 2 (slide 30):

Use line 11 as a guide: `let {id, volumeInfo: {title, authors, description}, saleInfo: {listPrice: {amount}}} = book;`

6. Display a description, price (amount), image and authors in the Book.js file

Copyright © 2020 Black Codher Bootcamp. All Rights Reserved. Do Not Redistribute.
50